На правах рукописи

Доренская Елизавета Александровна

МАТЕМАТИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДИНАМИЧЕСКОГО ПРОЕКТИРОВАНИЯ ТРАНСЛЯТОРА СО СПЕЦИАЛЬНОГО МЕТАЯЗЫКА ОПИСАНИЯ ЗАДАЧИ В ЯЗЫК ПРОГРАММИРОВАНИЯ ВЫСОКОГО УРОВНЯ

Специальность: 2.3.5. Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей

АВТОРЕФЕРАТ

диссертации на соискание учёной степени
кандидата технических наук

Воронеж — 2023

Работа выполнена в ФГБОУ ВО «Воронежский государственный технический университет».

Научный руководитель: **Кравец Олег Яковлевич,** доктор технических наук, профессор

Официальные оппоненты: **Иванова Галина Сергеевна,** доктор технических наук, профессор, ФГБОУ ВО «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)», профессор кафедры «Компьютерные системы и сети»;
Аникин Игорь Вячеславович, доктор технических наук, профессор, ФГБОУ ВО «Казанский национальный исследовательский технический университет им. А.Н.Туполева-КАИ», заведующий кафедрой информационной безопасности.

Ведущая организация: **ФГБОУ ВО «Рязанский государственный радиотехнический университет им. В.Ф. Уткина».**

Защита состоится «26» января 2024 года в 14:00 часов в конференц-зале на заседании диссертационного совета 24.2.286.04, созданного на базе ФГБОУ ВО «Воронежский государственный технический университет», по адресу: г. Воронеж, Московский просп., д. 14, ауд. 216.

С диссертацией можно ознакомиться в научно-технической библиотеке ФГБОУ ВО «Воронежский государственный технический университет» и на сайте www.cchgeu.ru.

Автореферат разослан «08» декабря 2023 г.

Ученый секретарь
диссертационного совета  Гусев Константин Юрьевич

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. Благодаря стремительному развитию информационных технологий возникла необходимость в снижении трудоёмкости написания программ и предотвращении появления ошибок. Большой вклад в развитие автоматизации и сокращение числа ошибок в программах внесли Holzmann G.J., Matej Balog, Alexander L. Gaunt, Аветисян А. И., Иванников В. П., Белеванцев А. А, Raghothaman, M. В теоретическом плане ряд методов автоматизации и предотвращения появления ошибок сводится к применению библиотек и специальных правил при написании программного кода. Разумной идеей кажется введение процедуры описания задачи с последующим преобразованием его в исполняемый код на алгоритмическом языке программирования высокого уровня.

Несмотря на то, что последние годы в области развития искусственного интеллекта случился сильный прогресс, он создан в основном с помощью нейронных сетей, которые разработаны по образу и подобию человеческого мышления. А это значит, скорее всего, будут совершать те же ошибки, что и человек. Ведущие исследователи в данном направлении: Andrew Ng, Yann Andre LeCun, Fei-Fei Li**.** Задача разработки кода с помощью нейронных сетей решается исследователями из Microsoft и Кембриджского университета. Чтобы устранить этот недостаток в области нейросетей, необходимо как можно большую часть работы по написанию кода перекладывать на ЭВМ. Это необходимо для предотвращения появления ошибок, вызванного наличием человеческого фактора, а также сокращения трудоёмкости разработки программного кода.

Реализация крупных проектов в области IT-технологий требует привлечения значительного количества квалифицированных специалистов. Сжатые сроки выполнения проектов приводят к необходимости сокращения времени на отладку программного обеспечения. Как правило, этого не удаётся добиться даже при увеличении числа программистов. Поэтому целесообразно провести оптимизацию процессов программирования за счёт разработки специализированной оболочки, генерирующей программные коды. В этом случае снимается противоречие, возникающее между разработчиком технического задания и программистом, создающим приложение.

Таким образом, актуальность темы диссертационного исследования продиктована необходимостью разработки математического и программного обеспечения автоматизации процесса программирования за счет введения процедуры описания задачи с последующим преобразованием его в исполняемый код на алгоритмическом языке программирования высокого уровня.

Тематика диссертационной работы соответствует научному направлению ФГБОУ ВО «Воронежский государственный технический университет» «Вычислительные комплексы и проблемно-ориентированные системы управления».

Целью работы является разработка математического и программного обеспечения динамически проектируемого транслятора со специального метаязыка описания задачи на язык высокого уровня для снижения трудоемкости генерации кода.

Задачи исследования. Для достижения поставленной цели необходимо решить следующие задачи:

1. Разработать синтаксис метаязыка описания задачи для генерации его с помощью транслятора в язык программирования высокого уровня.

2. Разработать алгоритм динамического выбора состава модулей для автоматического выбора способа решения задачи в рамках коллектива задач одного типа и сокращения количества модулей в базе данных.

3. Разработать алгоритм идентификации формализованных результатов внешней верификации программных модулей в базах данных для получения корректного результата в большинстве случаев без больших затрат ресурсов и сложных вычислений.

4. Разработать структуру динамического транслятора описания задачи в язык Perl, обеспечивающую автоматическое проектирование программы решения задачи на языке высокого уровня.

Объект исследования: способы автоматизации разработки программного обеспечения.

Предмет исследования: методы оценки снижения числа программных ошибок в коде, методы сокращения трудоёмкости написания программного кода.

Методы исследования: системный анализ, математическая статистика, использование специальных метрик и подходы, применяемые при разработке системного программного обеспечения.

Тематика работы соответствует следующим пунктам паспорта специальности 2.3.5. Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей: п.1 «Модели, методы и алгоритмы проектирования, анализа, трансформации, верификации и тестирования программ и программных систем»; п.2 «Языки программирования и системы программирования, семантика программ»; п.3 «Модели, методы, архитектуры, алгоритмы, языки и программные инструменты организации взаимодействия программ и программных систем».

Научная новизна работы. В диссертации получены следующие результаты, характеризующиеся научной новизной:

1. Синтаксис метаязыка описания задачи, отличающийся ориентацией на создание специальных описаний для генерации программного кода динамически конструируемым транслятором, обеспечивающий снижение трудоёмкости генерации кода на языке высокого уровня.

2. Алгоритм динамического выбора состава модулей для решения коллектива задач одного типа, отличающихся автоматическим учетом специфики задач и обеспечивающий рациональный состав модулей.

3. Алгоритм идентификации формализованных результатов внешней верификации программных модулей в базах данных, отличающийся учетом расстояний между текстовыми элементами результатов верификации и их весов в специальной базе данных для подбора корректной семантики, обеспечивающий корректный результат в более чем 95% случаев.

4. Структура динамического транслятора описания задачи в язык Perl, отличающаяся автоматическим учетом контекста задачи и обеспечивающая проектирование программы решения задачи на языке высокого уровня.

Теоретическая и практическая значимость исследования заключается в разработке математического и программного обеспечения динамически конструируемого транслятора со специального метаязыка описания задачи на язык программирования высокого уровня с целью снижения трудоемкости генерации кода и сокращения числа ошибок, а также алгоритма идентификации формализованных результатов внешней верификации программных модулей.

Теоретические результаты работы могут быть использованы в проектных и научно-исследовательских организациях, для снижения трудоёмкости создания программного кода, благодаря чему его могут создавать высококвалифицированные специалисты в какой-либо области наук не владеющие языками программирования. Алгоритм идентификации формализованных результатов внешней верификации программных модулей может использоваться для распознавания смысла специализированных текстовых описаний с помощью ЭВМ.

Положения, выносимые на защиту.

1. Синтаксис метаязыка описания задачи обеспечивает снижение трудоёмкости генерации кода на языке высокого уровня.

2. Алгоритм динамического выбора состава модулей для решения коллектива задач одного типа позволяет обеспечить рациональный состав модулей.

3. Алгоритм идентификации формализованных результатов внешней верификации программных модулей в базах данных позволяет получать корректный результат в более чем 95% случаев.

4. Структура динамического транслятора описания задачи в язык Perl, позволяет автоматически генерировать программы для решения задач на языке высокого уровня.

Результаты внедрения. Основные результаты внедрены в группе технической поддержки компьютерных линий связи Национального исследовательского центра «Курчатовский институт» Института теоретической и экспериментальной физики имени А. И. Алиханова (ИТЭФ), для поддержки безопасности веб-серверов.

Апробация работы. Основные положения диссертационной работы докладывались и обсуждались на следующих конференциях: 60-й всероссийской научной конференция МФТИ (Москва, 2017 г.); 8-й Международной конференции «Распределенные вычисления и GRID-технологии в науке и образовании». (Дубна, 2018); Молодежной конференции по теоретической и экспериментальной физике (Москва, 2018-2021 гг.); XV и XVII Международной научно-практической конференции «Современные информационные техноло-

гии и ИТ-образование» (Москва, 2020 и 2022гг.); Всероссийской научно-практической конференции «Решение» (г. Березники, 2022 г.), а также на научных семинарах ОИЯИ, МИФИ, НИЦ «Курчатовский институт», ИСП РАН (2021-2022 гг.)

Достоверность результатов обусловлена корректным использованием теоретических методов исследования и подтверждена результатами сравнительного анализа данных вычислительных и натурных экспериментов.

Публикации. По результатам диссертационного исследования опубликовано 19 научных работ (12 – без соавторов), в том числе 18 – в изданиях, рекомендованных ВАК РФ (из них 1 – в издании Scopus, 1 – патент на изобретение и 11 свидетельств о регистрации программы для ЭВМ). В работах, опубликованных в соавторстве и приведенных в конце автореферата, лично автором получены следующие результаты: [1,6] – алгоритм динамического выбора состава модулей для решения коллектива задач одного типа; [2,3,7] –алгоритм идентификации формализованных результатов внешней верификации программных модулей в базах данных; [4,5] - синтаксис языка описания задачи (CDTL), структура транслятора описания задачи на языке CDTL в PERL и базы данных алгоритмов.

Структура и объем работы. Диссертационная работа состоит из введения, четырех глав, заключения, списка литературы из 135 наименований и приложения. Текст диссертации изложен на 146 страницах.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во введении обоснована актуальность исследования, сформулированы его цель и задачи, научная новизна и практическая значимость полученных результатов, приведены сведения об апробации и внедрении работы.

В первой главе исследуются особенности обнаружения ошибок в программном обеспечении и методов контекстного анализа, способы автоматизации программирования. Описаны последствия возникновения программных ошибок. Отмечено, что рассчитать сокращение количества ошибок в программах и трудоёмкость их написания можно с помощью специальных метрик. Дан небольшой обзор таких метрик.

По итогу анализа данных метрик для оценки сокращения количества ошибок были применены: простая интуитивная модель и метрика Холстеда. Простая интуитивная модель была выбрана потому что имелись ресурсы, необходимые для реализации расчётов с её использованием, и можно было получить нужную оценку по количеству ошибок. Помимо этого, применить её было проще чем другие, упомянутые выше, модели. Метрика Холстеда была выбрана потому что она подходит для решаемой задачи и с помощью неё довольно просто сделать нужные расчёты. Результат данного анализа потребовал формализации данных задач, а также алгоритмизации их решения с учетом особенностей, отраженных на рис. 1.

Рис. 1. Дизайн исследования

Были сформулированы цель и задачи исследования.

Вторая глава посвящена разработке специального метаязыка описания задачи и алгоритма динамического выбора состава модулей для решения коллектива задач одного типа.

Для формирования описания задачи был разработан специальный метаязык для сокращения числа ошибок и снижения трудоёмкости написания программного кода, который получил название Creating Description Task Language (CDTL).

Внутри описания задачи на CDTL может содержаться описание субзадачи. Начало описания задачи отмечается знаком <#, а завершение - #>. Начальный и завершающий разделители отделяются от описания задачи пробелом. Например, <# name="filter.pl"; .... #>. Имя описанной программы задается оператором name, здесь "filter.pl" - имя будущей программы.

Сокращенное CDTL-описание скрипта filter.pl предоставлено ниже:

<# name="filter.pl"; librs "IO::File, Fcntl, Socket Time::Local, Time::localtime"; OUT{} = all records from IN, not_containing elements from @list_of_exceptions; #>;

Скрипт filter.pl читает журнальный файл Apache access_log и удаляет из него вспомогательные запросы графических файлов, запросы со стороны поисковых серверов и т.д., что в разы ускоряет последующую работу программного пакета. Метазык CDTL состоит из операторов, атрибутов, объектов, параметров, описаний, переменных и массивов (табл. 1).

Таблица 1. Примеры элементов метаязыка CDTL

Операторы	**Атрибуты**	**Объекты**	**Пара-**	**Описания**	**Переменны**	**Массивы**
develop; create; write; read; convert;	fast; double-precision; by polinomial	program; file; files DB; table;	label; parameters accuracy; dt;	Что программа должна делать: compute system_control;	Задаются IN= OUT = $=	Задаются @имя

Для того, чтобы оценить насколько уменьшается описание, написанное человеком на языке CDTL, по сравнению с кодом на языке Perl был вычислен коэффициент сжатия. Он был рассчитан для более чем 10 описаний на языке CDTL и аналогичных программ на Perl. Данный коэффициент показывает во сколько раз сокращается текст при использовании CDTL. Была построена зависимость вероятности коэффициента сжатия текста при использовании CDTL (см. рис. 2).

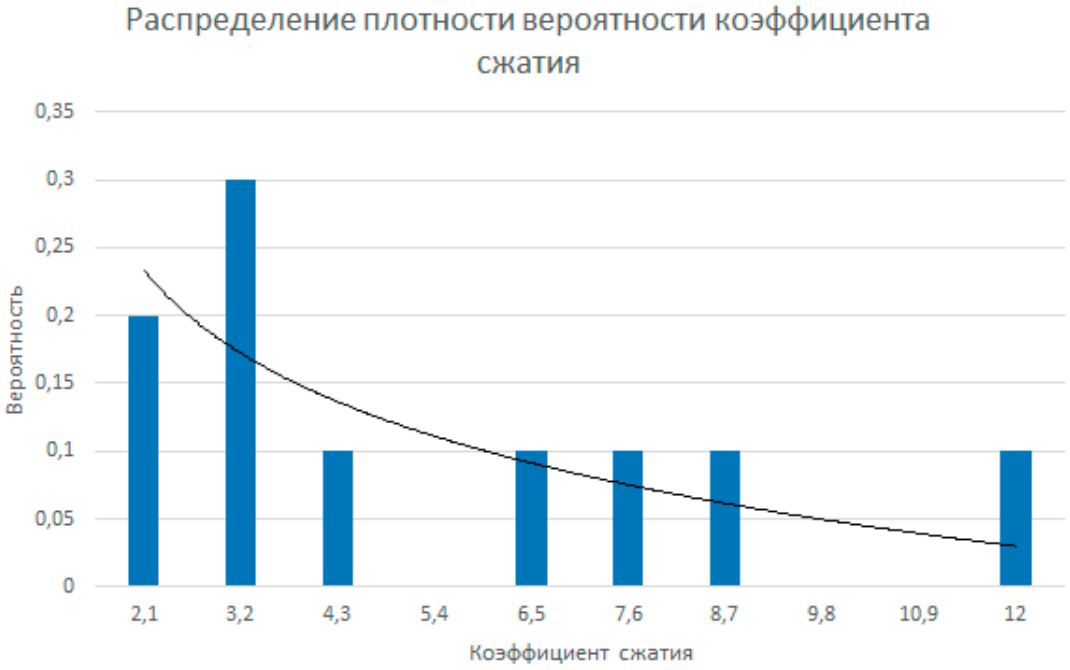

Рис. 2. Зависимость вероятности коэффициента сжатия текста при использовании CDTL

Минимальный коэффициент сжатия при использовании CDTL в данной выборке равен 2,1. Максимальный коэффициент = 12. Для данных значений

была рассчитана дисперсия, которая оказалась равна 9,8. Математическое ожидание коэффициента сжатия текста описания CDTL по сравнению с языком PERL составляет 5,3±3,1. Данный результат является довольно значительным и это показывает, что при использовании CDTL трудоёмкость написания программы существенно снижается. Среднеквадратичное отклонение коэффициента сжатия σ получилось довольно небольшим – 3,1.

Таким образом, разработан синтаксис метаязыка описания задачи, отличающийся ориентацией на создание специальных описаний для генерации программного кода динамически конструируемым транслятором, обеспечивающий снижение трудоёмкости генерации кода на языке высокого уровня.

Алгоритм динамического выбора состава модулей для решения коллектива задач одного типа был разработан с целью объединения блока схожих программ в один модуль для сокращения их количества и автоматизации выбора способа решения задачи. Данный алгоритм использует вычисление определенных функций и проверку результата этого расчета. По результатам проверки принимается решение о выполнении того или иного фрагмента кода из числа имеющихся. Алгоритм модифицируется так, чтобы отвечать требованиям списков входных и выходных параметров (см. рис. 3).

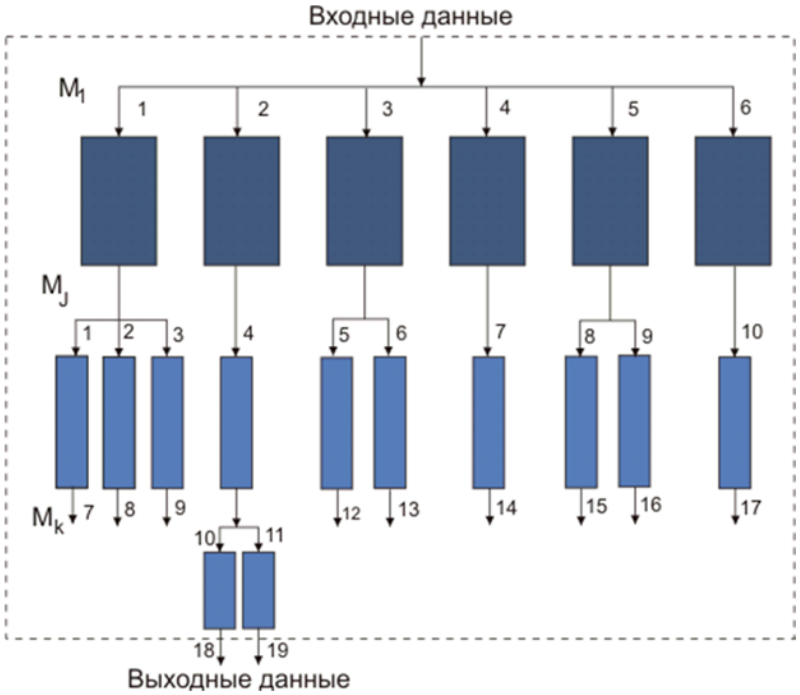

Рис. 3. Схема реализации программы типа m_gnuplot

Программа может формироваться на основе входных параметров (параметрическое управление), например, как в случае модификации кода для многовариантной программы, m_gnuplot (рис. 3). На нём же показан пример программы, управляемой входными параметрами. Программа служит для формирования графических распределений с привлечением библиотечных функций gnuplot (библиотека Chart-Graph-3.2, которая содержит в себе несколько модулей). Языком программирования для написания кода был выбран Perl. Алгоритм динамического выбора состава модулей для автоматического выбора способа решения задачи в рамках коллектива задач одного типа и сокращения количества модулей в базе данных представлен на рис. 4.

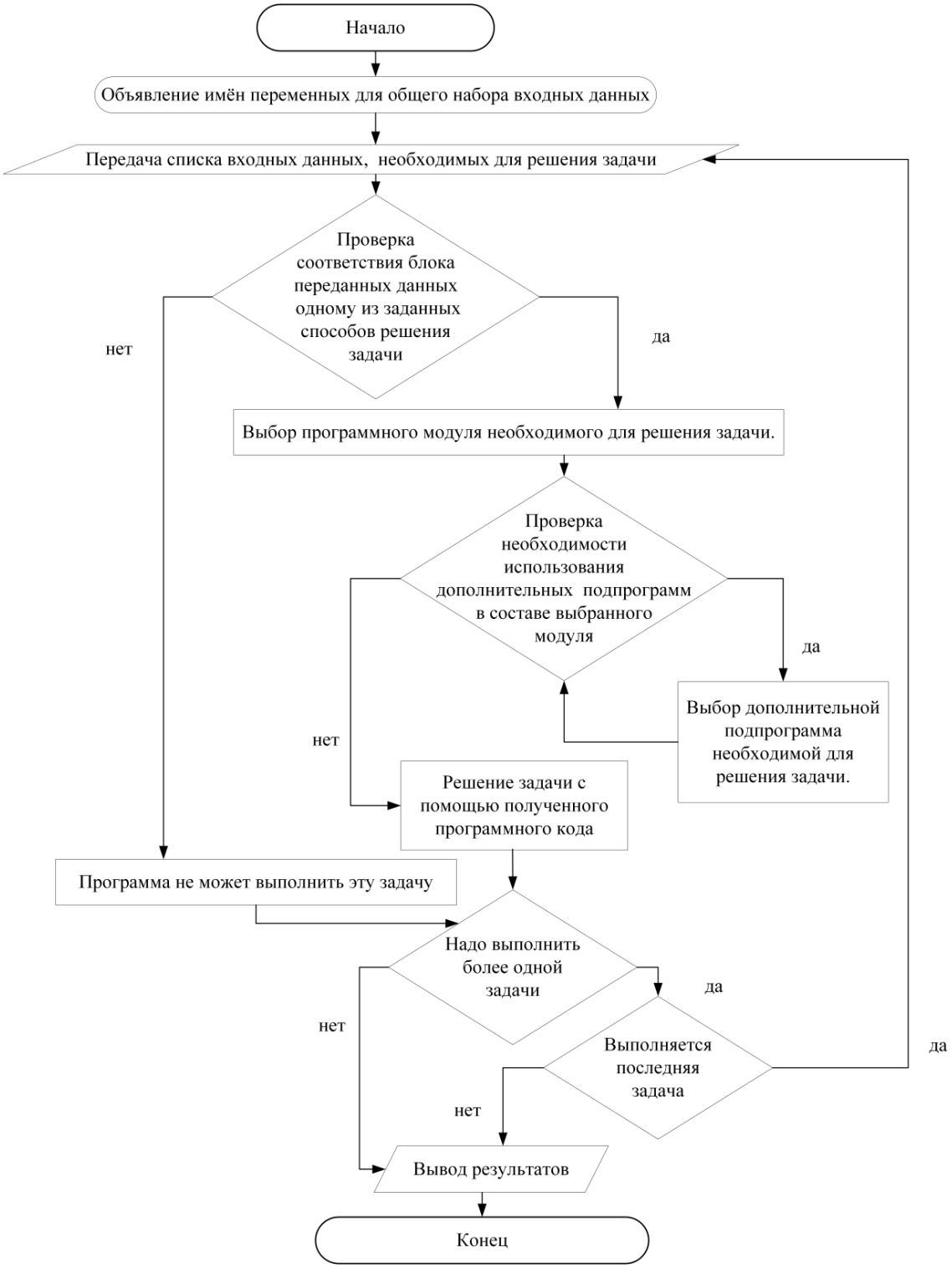

Рис. 4. Структурная схема алгоритма динамического выбора состава модулей для автоматического выбора способа решения задачи

Таким образом разработан алгоритм динамического выбора состава модулей для автоматического выбора способа решения задачи в рамках коллектива задач одного типа и сокращения количества модулей в базе данных.

Третья глава посвящена алгоритму идентификации формализованных результатов внешней верификации программных модулей в базах данных и фазам работы транслятора из CDTL в код языка Perl.

При разработке пакета программ для защиты Web-сервера был создан банк алгоритмов. В банке алгоритмов находятся модули, меняя входные данные которых, можно варьировать их работу. Это увеличивает область их применения и уменьшает необходимое количество скриптов. В отличие от депозитария Cpan, код модулей данного банка алгоритмов полностью доступен для редактирования, т. к. они написаны на интерпретируемом языке программирования Perl. И в отличие от библиотек Cpan их не надо специально устанавливать. Для хранения информации о программных модулях, находящихся в банке алгоритмов, была использована таблица базы данных MySQL.

Способ определения контекста слов планируется применять для внешней верификации программных модулей в базах данных алгоритмов. Данный анализ необходим для выявления и устранения ошибок в программах, находящихся в банке алгоритмов.

Так как отзывы написаны на естественном языке, нужна автоматизированная система его анализа. При таком анализе часто бывает нужно определять контекстные значения отдельных слов и фрагментов текста. Смысл текста отзывов на модули банка алгоритмов программой распознаётся с помощью ключевых слов. А для определения контекста ключевых слов использован алгоритм, описанный в данной диссертационной работе. Для анализа отзывов была создана специальная программа.

Данный алгоритм заключается в том, что контекстное значение слова зависит от расстояния L между этим словом и другими словами, задающими контекст. Расстояние между словами определяется числом слов N, размещенных между ними (L=N+1). Предполагалось, что контекст конкретного слова можно определить по положению некоторых семантически связанных с ним слов, содержащихся в тексте.

Корневое слово W1 может иметь два или более значений, зависящих от контекста и определяемых словами W2. Слова W2 могут и отсутствовать в тексте документа. Контекстное значение слова W1 в этом случае может определяться семантически связанными с ним словами W3.

Варианты семантических сетей показаны на рис. 5. Вариант A предполагает наличие в тексте документа корневого слова W1, которое может иметь разные контекстные значения, определяемые словами W2. Некоторые слова-значения W2 могут в документе отсутствовать (рис. 5B).

Предполагается, что каждому из слов W2 соответствует некоторое число слов W3 (слова-характеристики), именно они и определяют выбор контекстного значения слова W1. Секция рис. 5C иллюстрирует вариант оценки контекста документа в отсутствии слова W2. В таблице 2 представлены варианты семантических связей в тексте, подобные изображённым на рис. 5. В таблицу заносятся только слова, имеющие два или более контекстных значений (W2).

В первой колонке таблицы размещаются слова, которые могут иметь несколько контекстных значений (корневые слова - W1) и могут также определять контекст документа в целом. Во второй колонке (W2) помещаются слова, которые обозначают возможные контекстные значения слов из первой колонки. В третьей колонке (W3) записаны слова, конкретизирующие значения слов из второй и первой колонки. Слова из этих трех колонок образуют древовидный граф. Значения веса M относятся к словам из третьей колонки таблицы.

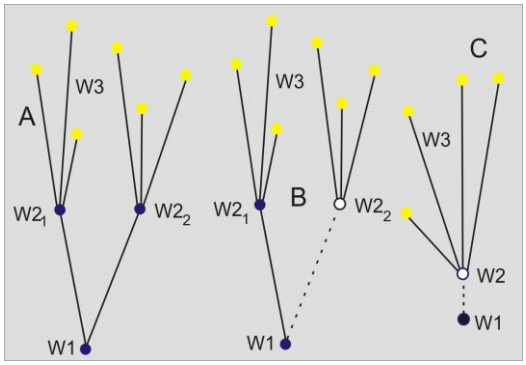

Рис. 5. Варианты семантических связей в тексте

Таблица 2. Фрагмент таблицы корневых слов (W1=”Программа”), слов-значений (W2) и слов-характеристик (W3)

Корневое слово W1	**Слова-значения W2**	**Слова-характеристики (W3)**	**Метрика [M]**
Программа	Компьютер	программирование	**70**
		отладка	**60**
		тестирование	**40**
		подпрограмма	**30**
		объект	**15**
		файл	**26**
	Обучение	пособие	**45**
		преподаватель	**50**
		учащийся	**95**

На рис. 6 изображён пример фрагмента семантической сети в виде графа, иллюстрирующего таблицу 2. На нём отображены слова из первой (W1), второй (W2) и третьей (W3) колонок данной таблицы. В самом его верху находится корневое слово W1 «программа». От него идут ветвления к словам W2, которые обозначают контекстный смысл слова W1 «программа». Метрика характеризует близость слова к контекстному значению, с которым оно связано и обозначается буквой M. Слова во второй колонке всегда имеют метрику равную 100. Поэтому для слов «компьютер» и «обучение» M=100. Далее от них идут ветвления графа к словам из третьей колонки W3. Это все прочие слова в таблице. Под каждым из этих слов указана его метрика (<100) и они сильно влияют на контекст, определяемый для слова W1.

На рис. 6 отображен фрагмент схемы, поскольку контекстных смыслов у слова «программа» может быть много. Существует также много слов-характеристик, помимо указанных в колонке W3 таблицы 2. Графы такого вида и являются основой семантической сети, используемой для оценки контекстного значения слов и документов.

Можно предположить, что чем ближе слово-характеристика к слову из вышестоящей вершины графа, тем с большей вероятностью оно определяет контекст этого слова. Наличие слова из третьей колонки, размещенного в тексте ближе к слову из второй колонки, должно влиять на выбор контекстного значения слова сильнее, чем в случае слов, размещенных дальше.

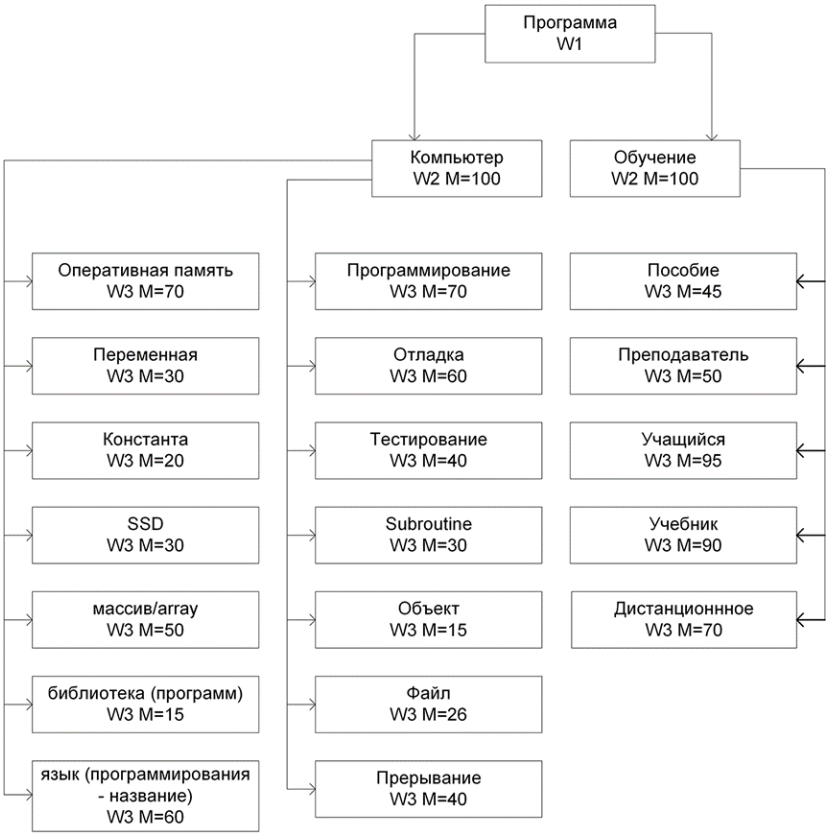

Рис. 6. Структурная схема фрагмента семантической сети в виде графа, иллюстрирующего таблицу 2

Одним из возможных методов оценки контекстного значения слова может быть формула 1. После того как положение слов W1, W2 и W3 определено, производится вычисление суммы C**.**

$$\mathbf{C_{k,n}} = \sum_{i=1}^{m} (M_i \times f(L_i)), \quad (1)$$

где C – мера, определяющая контекстное значение слова W1, L - расстояние между словом, например, «компьютер» и «отладка» (см. табл. 2), M_i - метрика слова-характеристики W3 (M=1÷100), m - число семантически связанных слов W3 $f(L_i)$ - весовая функция от L_i, i - номер встретившегося слова из колонки 3. В простейшем случае $f(L_i) = 1/L_i$**,** а для небольших документов $f(L_i) = 1$**.** L определяется числом слов N, размещенных между словом W2 и одним из слов W3 (L=N+1). Весовая функция $f(L_i)$ нужна для ослабления влияния удаленных слов на оценку контекстного значения слова W1**.** Если в тексте присутствует две или более копий слова W2, формула 1 может быть модифицирована.

Индекс k для C определяет - к какому из возможных значений W2 относится данная мера (k=1,..n), см. вторую колонку таблицы 2, n – число возможных значений слова W1 (чаще всего n=2÷3). Значение слова W2 с большим значением C в контекстном смысле считается предпочтительным. Значения M_i выбираются при настройке с использованием тестовых документов.

Механизм распознавания контекста моделировался по методу Монте-Карло. Предполагалось, что в документе имеется N слов. При моделировании считалось, что положение слов в документе имеет постоянную плотность

вероятности (слова размещены в документе статистически равномерно, что не всегда справедливо). Для анализа в документ заносились случайным образом слова «программа» и слова-характеристики.

Вероятность p, например, получения определенного значения C может быть оценена на основе распределения плотности вероятности. Вероятность P получения C=9,12 равна 0,06, при этом вероятность C=2,38 <0,001.

В случае использования неравенства Чебышева имеем:

$$p(|x-\overline{C}-| \geq \Delta C) \leq (\sigma^2/(\Delta C)^2). \quad (2)$$

Это неравенство определяет верхнюю границу вероятности того, что разность случайной величины x и $\overline{C}$ превышает определенный порог ΔC для произвольного распределения с дисперсией σ^2 и средним значением $\overline{C}$.

Например, в документе «SET и другие системы осуществления платежей» можно программно вычислить значение $\overline{C}$ = 9,12 и σ =14,0 для слова «компьютер». При этом для слова «реализация» (программы) получаем $\overline{C}$= 2,38, а σ=4,73. ΔC=9,12-2,38=6,74 (разница между математическими ожиданиями взятых нами распределений).

Неравенство Чебышева для этого случая имеет вид:

$$P(|X-2,38|\geq(9,12-2,38)) \leq 4,73^2/(9,12-2,38)^2$$

$$P(|X-2,38|\geq 6,74) \leq 4,73^2/6,74^2.$$

Исходя из этого получается что:

$$P(|X-2,38|\geq 6,74) \leq 0,49.$$

Это вполне согласуется с оценкой по плотности вероятностей при моделировании и подтверждает корректность распознавания контекста. Неравенство Чебышева удобно использовать, когда число слов W1 в документе достаточно велико. Испытание алгоритма на полутора десятках текстовых файлах показало, что достоверная оценка получается в более чем 95% случаев.

Если в документе содержится более одного образца слова-значения W2, соответствующего данному W1, то формула расчета контекстных значений W1 должна быть изменена.

$$\mathbf{C_{k,n}} = \sum_{j=1}^{K} \frac{\sum_{i=1}^{m}(M_i \times f(L_i))}{D_j}, \quad (3)$$

где D_j характеризует расстояние от данного W1 до одного из слов W2 с номером j; K – число конкретных слов-значений W2 в документе. Если в документе слово W2=”code” встречается 4 раза, то K=4, $D_j = |L_j - L_0| + 1$, L_0 – минимальное расстояние от W1 до данного конкретного W2.

Для того чтобы проверить функциональность данной формулы, были написаны специальные программы и по одной из них было получено авторское свидетельство. По итогу этих работ был разработан алгоритм идентификации формализованных результатов внешней верификации программных модулей. Данный алгоритм представлен на рис. 7.

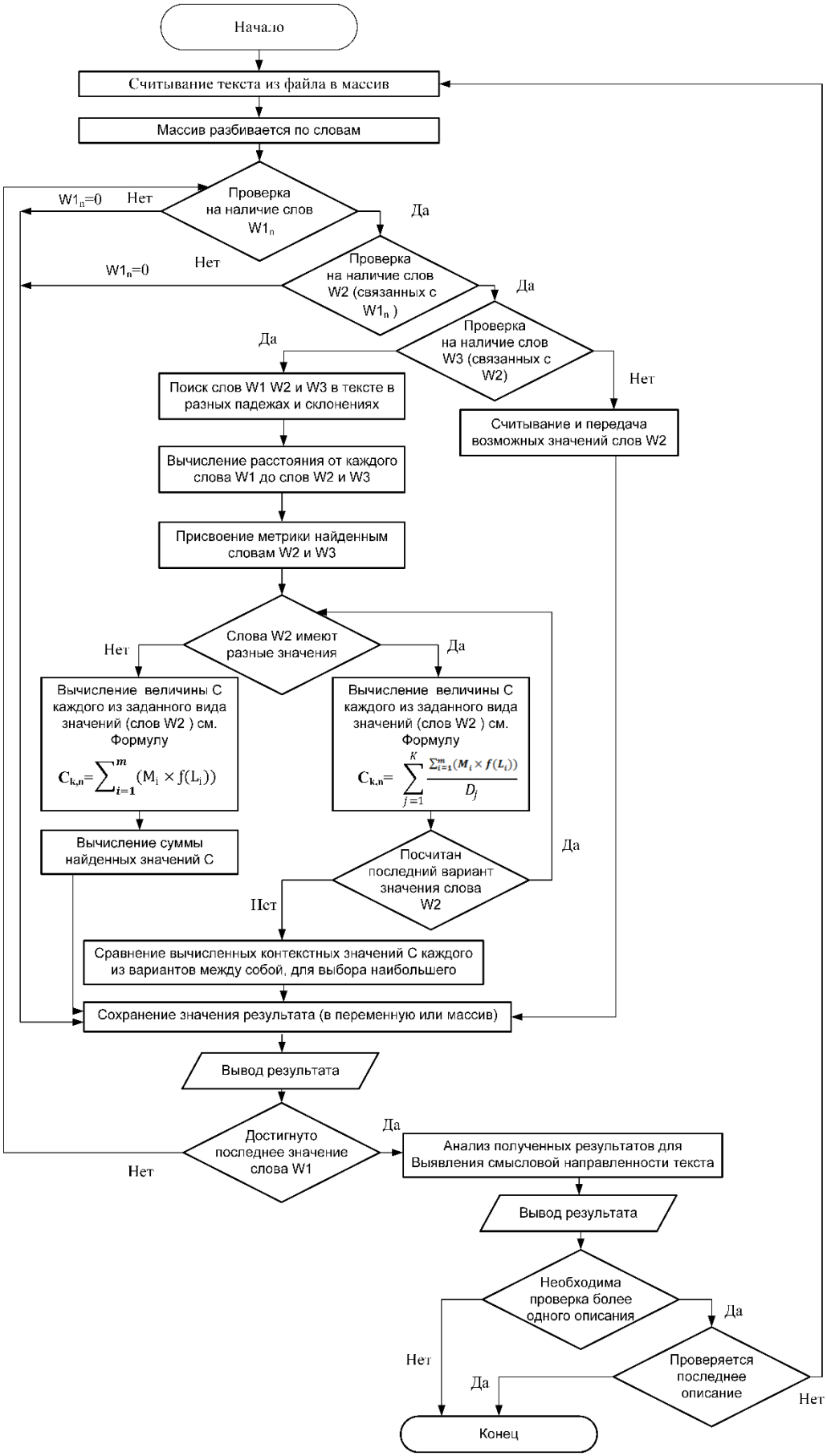

Рис. 7. Структурная схема алгоритма идентификации формализованных результатов внешней верификации программных модулей в базах данных

Таким образом, разработан алгоритм идентификации формализованных результатов внешней верификации программных модулей в базах данных, отличающийся учетом расстояний между текстовыми элементами результатов верификации и их весов в специальной базе данных для подбора корректной семантики, обеспечивающий корректный результат в более чем 95% случаев.

Работа транслятора из CDTL в Perl подразделяется на 2 фазы: фазу анализа (parser) и фазу синтеза.

На фазе анализа сначала формируется каталог с именем <имя> (имя каталога совпадает с именем файла описания задачи, заданным оператором name). Исходный текст файла <имя>.pdl читается построчно и формируются таблицы: operators, descriptions, variables, arrays, objects, attributes, parameters и т. д. Все перечисленные файлы укладываются в каталог <имя>. Транслятор может распознавать вставки на языке Perl, добавленные в описание задачи. Они должны быть выделены специальными тегами (<perl> текст программы на Perl </perl>). Они нужны для случая, когда в банке алгоритмов и библиотеке примитивов нет нужных модулей. При выявлении ошибок формируется файл errors.pdl.

Работа на фазе синтеза в данном трансляторе существенно отличается от аналогичной фазы других трансляторов. На ней последовательно рассматриваются описания задач, и с использованием имеющихся таблиц для каждого описания формируется текст программы на языке Perl.

Учитывая модульность метаязыка описания задач, транслятор должен быть также модульным.

Фаза синтеза может работать 3-мя способами:

1. Если в описании присутствует хотя бы один оператор действия, то проводится поиск модулей в банке алгоритмов (хранилище описаний алгоритмов).
2. Если модуль в банке алгоритмов не найден, то в описании задачи должны присутствовать имена примитивов. Тогда код программы генерируется из примитивов.
3. Если при тех же условиях, что в пункте 2, в описании задачи нет имен примитивов, то модуль придётся частично или полностью писать самому программисту.

В первом случае, как уже было сказано, необходимо провести поиск в банке алгоритмов.

Во втором случае используются примитивы. Примитив – программный модуль, имеющий одну и только одну алгоритмическую функцию, например, вычисление факториала или анализ записей журнального файла по какому-то параметру. Имена примитивов в CDTL начинаются с **p_**(например **p_ip_to_num)**.

Там, где в описании помещено имя примитива, в код вставляется обращение к примитиву. Там, где в описании помещено имя примитива, в код

вставляется обращение к примитиву. В этом варианте в описании задачи содержатся элементы алгоритма.

В четвёртой главе представлены результаты апробации метаязыка описания задач с помощью транслятора из CDTL в Perl, описаны программные модули, используемые при тестировании метаязыка описания задач. Произведена экспериментальная оценка алгоритма, используемого для внешней верификации программных модулей в базах данных.

Для апробации метаязыка CDTL был создан специальный транслятор (см. рис. 8.).

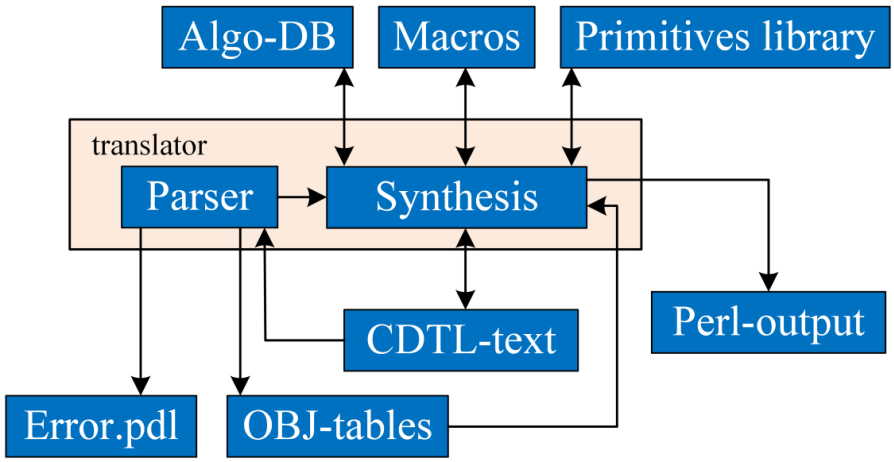

Рис. 8. Структурная схема динамического транслятора описания задачи в язык Perl

CDTL-text - текст описания задачи на CDTL. ALGO-DB - банк описаний алгоритмов. Perl-output - программа, формируемая транслятором (результат). Error.pdl - журнал ошибок, выявленных парсером. OBJ-tables - таблицы результатов анализа текста описания. Primitives library – библиотека примитивов. Macros - библиотек макросов.

Работа транслятора подразделяется на 2 фазы: фазу анализа (парсер) и фазу синтеза (генерация исполняемого кода). Скрипты error.pl, pars_access_2.pl и secur_cur.pl синтезируются первым способом (см. главу 3), а скрипты filter.pl и proc.pl - вторым. В функционал данного пакета программ входит блокировка IP-адресов, с которых программа замечает много подозрительных запросов.

Пакет программ защиты Web-сервера, сгенерированный транслятором для его апробации мониторирует входящий и исходящий информационный трафик, детектирует атаки переполнения буфера, попытки DoS-атак и path traversal, а также случаи использования нелегальных HTTP-методов (put, delete и пр.) и некоторых других атак.

Таким образом разработана структура динамического транслятора описания задачи в язык Perl, отличающаяся автоматическим учетом контекста задачи и обеспечивающая проектирование программы решения задачи на языке высокого уровня.

Для оценки сокращения числа программных ошибок, которое должно произойти благодаря применению CDTL, удобнее всего использовать метрику Холстеда и простую интуитивную модель. Результаты расчетов по данным методикам представлены в таблице 3.

Таблица 3. Расчёт сокращения числа ошибок с помощью метрики Холстеда и простой интуитивной модели в пакете программ защиты Web - сервера при использовании CDTL, проводился для всех программ пакета (суммарно). В последней колонке показано, на сколько процентов сокращается количество ошибок при использовании CDTL

Количество строк кода		Объём в битах		Приблизительное количество ошибок, шт.		Сокращение ошибок в % при использовании CDTL	
в описании задачи на CDTL	в программе на Perl	Описания задачи на CDTL	в программе на Perl	в описании задачи на CDTL	в программе на Perl	Метрика Холстеда	Простая интуитивная модель
52	335	29600	105064	9,87	35,02	71,8	86,7

Сложность создания программ защиты Web- сервера на языке CDTL и на Perl, была вычислена с помощью 2-х метрик разного вида. Сложность программ на Perl по метрике Холстеда равна 104,92, что выше аналогичного показателя программ CDTL (23,6) на 77,5%. Сложность программ на Perl по метрике Джилба равна 153, что выше аналогичного показателя программ CDTL (4) на 97,4%. Из этих показателей следует, что описание задачи на CDTL написать проще, чем код на Perl.

ЗАКЛЮЧЕНИЕ

В процессе выполнения диссертационного исследования получены следующие основные результаты:

1. Разработан синтаксис метаязыка описания задачи, обеспечивающий снижение трудоёмкости генерации кода на языке высокого уровня.
2. Разработан алгоритм динамического выбора состава модулей для решения коллектива задач одного типа, обеспечивающий рациональный состав модулей.
3. Предложен и проверен алгоритм идентификации формализованных результатов внешней верификации программных модулей в базах данных, позволяющий в более чем 95% случаев получить корректный результат, при котором не требуется сложных вычислений и больших затрат ресурсов.
4. Создан прототип динамического транслятора описания задачи в язык Perl, обеспечивающий проектирование программы решения задачи на языке высокого уровня.
5. Разработанное программное обеспечение защищено патентом РФ, получено 11 свидетельств о государственной регистрации в Роспатенте.

Основные результаты диссертации опубликованы в следующих работах:

Публикации в изданиях, рекомендуемых ВАК РФ

1. Доренская Е. А. О технологии программирования, ориентированной на минимизацию программных ошибок / Е.А. Доренская, Ю.А. Семенов // Современные информационные технологии и ИТ-образование. – 2017. –Т. 13. – № 2. – С. 50-56.
2. Доренская Е. А. Метод определения контекстных значений слов и документов / Е.А. Доренская, Ю.А. Семенов // Современные информационные технологии и ИТ-образование. – 2018. – Т. 14. – №4. – С. 896-902.
3. Доренская Е. А. Улучшенный алгоритм вычисления контекстного значения слов в тексте / Доренская Е. А., Семенов Ю.А. // Современные информационные технологии и ИТ-образование. – 2019. – Т. 15. – №4. – С. 954-960.
4. Доренская Е. А. Язык описания проблемы и исследование его возможностей/ Е.А. Доренская, Ю.А. Семенов, А.А. Куликовская // Современные информационные технологии и ИТ-образование. – 2020. – Т. 16 – №3. – С. 653-663.
5. Куликовская А. А. Количественные характеристики безопасности программ / А. А. Куликовская, Е.А. Доренская, Ю.А. Семенов // Современные информационные технологии и ИТ-образование. – 2022. – Т. 18. – №4. – С. 855-860.

Публикация в издании, индексируемом в Scopus

6. Dorenskaya E.A., New methods of minimizing the errors in the software / E.A. Dorenskaya, Y.A. Semenov // Proceedings of the VIII International Conference «Distributed Computing and Grid-technologies in Science and Education» (GRID 2018), Dubna, Moscow region, Russia. – 2018. - Vol 2267, Pp. 150-154

Патент на изобретение

7. Доренская Е. А., Семенов Ю.А. Способ определения контекста слова и текстового файла. Патент на изобретение № 2685044 от 16.04.2019. М.: ФИПС, 2019.

Свидетельства о государственной регистрации программ для ЭВМ

8. Доренская Е. А. Формирование регулярного выражения посредством диалога. – Свидетельство о государственной регистрации программы для ЭВМ № 2017619074 от 15.08.2017. М: ФИПС, 2017.
9. Доренская Е. А. Организация запросов для генерации регулярных выражений с помощью специальных слов. – Свидетельство о государственной регистрации программы для ЭВМ № 2017663254 от 28.11.2017. М: ФИПС, 2017.
10. Доренская Е. А. Автоматическая генерация программ для построения графиков с помощью библиотеки gnuplot. – Свидетельство о государственной регистрации программы для ЭВМ № 2017663858 от 13.12.2017. М: ФИПС, 2017.

11. Доренская Е. А. Расчёт контекстного значения заданного слова в текстовом файле. – Свидетельство о государственной регистрации программы для ЭВМ № 2018615758 от 16.05.2018. М: ФИПС, 2018.
12. Доренская Е. А. Программа для защиты от перегрева ядер процессора V 1.0. – Свидетельство о государственной регистрации программы для ЭВМ № 2019616336 от 22.05.2019. М: ФИПС, 2019.
13. Доренская Е. А. “Wordcontext” v1.0 для расчета контекстного значения слова с учетом расстояний. – Свидетельство о государственной регистрации программы для ЭВМ № 2019660440 от 06.08.2019. М: ФИПС, 2019.
14. Доренская Е. А. Программа “Loader v 1.0” для загрузки описаний в базу данных алгоритмов. – Свидетельство о государственной регистрации программы для ЭВМ № 2019663880 от 24.10.2019. М: ФИПС, 2019.
15. Доренская Е. А. Универсальный конфигуратор ПО для пакета программ защиты Web-сервера. – Свидетельство о государственной регистрации программы для ЭВМ № 2020616122 от 10.06.2020. М: ФИПС, 2020.
16. Доренская Е. А. Анализ описаний PDL и трансляция с помощью примитивов. – Свидетельство о государственной регистрации программы для ЭВМ № 2020662902 от 20.10.2020. М: ФИПС, 2020.
17. Доренская Е. А. Программа “Databcont1” определение контекста слов с помощью специальной базы данных. – Свидетельство о государственной регистрации программы для ЭВМ № 2021666997 от 22.10.2021. М: ФИПС, 2021.
18. Доренская Е. А. Программа оценки качества программных модулей банка алгоритмов. – Свидетельство о государственной регистрации программы для ЭВМ, № 2023619401 от 11.05.2023. М: ФИПС, 2023.

Статьи и материалы конференций

19. Доренская Е. А. Методы создания изображений и программ с использованием описаний и функций // Решение: матер. 11-й Всеросс. НПК. – Пермь: Изд-во Перм. нац. исслед. политехн. ун-та. – 2022. – С. 206-209.

Подписано в печать 24.11.2023.
Формат 60x84/16. Бумага для множительных аппаратов.
Усл. печ. л. 1,0. Тираж 80 экз. Заказ №683
ФГБОУ ВО «Воронежский государственный технический университет»
394026 Воронеж, Московский просп., 14